Serie de instrucciones Algorítmicas

Rutina de cocinar;

1° Abrir la nevera..

2° sacar las verduras

3° cerrar la nevera

4° poner sobre la mesa las verduras

5° sacar y colocar los utensilios (/tabla, ollas, cuchillo/)

6° picar y sazonar las verduras

7° esperar a que se cocine la comida

8° emplear y/o colocar la mesa

9° hacer el jugo (/elegir entre las opciones //

10° servir la comida

11° colocar los utensilios (/platos, cucharas, tenedores etc./)

12° disfrutar la comida

13° reposar la comida (/tiempo estimado 15 minutos/)

14° recoger los platos y utensilios

15° lavar los platos

16° recoger platos limpios

17° colocar dentro de la lacena

Evidencia 3.3.1.1:

1° Lenguaje de Programación: Es un lenguaje en código mecanizado el cual se usa para dar a entender funciones entre el computador (maquina) y el usuario (recurso humano) para la interacción e iteración, en el cual se da entender conceptos y funciones en determinada lengua tipo: lenguaje C, Python, PHP etc.

En términos generales, un lenguaje de programación se conforma de una serie de símbolos y reglas de sintaxis y semántica que definen la estructura general y le dan un significado a sus elementos y expresiones.

(<https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html> )

2° Algoritmo: Es un conjunto ordenado y finito de operaciones que permite hallar la solución del problema, Sin embargo, la verdadera utilidad de un algoritmo es que el resultado sea lo más preciso y determinado posible, para así, poder utilizar el producto que ha generado. Este tipo de operaciones se dan en situaciones en las que las tareas puedan ser programas y los escenarios puedan acotarse a un abanico de posibilidades que el creador del algoritmo conozca bien.

Por otra parte, los algoritmos son objetos principales de entrada y salida fijas de pasos o convirtiéndose valores independientes con o sin maquina dependiendo del lenguaje; encontrándose no solo en matemáticas y la informática sino, en la vida cotidiana, desde las luces de los semáforos hasta la función principal de un ascensor

( <https://www.ionos.es/digitalguide/online-marketing/analisis-web/que-es-un-algoritmo/> )

3° Lenguaje de Maquina: Es un sistema de código interpretable por un circuito microprogramable compuesto por un conjunto de instrucciones conectadas por cadenas de secuencias con eventuales cambios de flujo causadas por el propio programa o agentes externos; por consiguiente, los circuitos microprogramables son digitales, lo cual significa que solo laboran con dos dígitos principales y únicos en niveles de tensión 0 y 1, Esto permite el empleo de las teorías del [álgebra booleana](https://es.wikipedia.org/wiki/%C3%81lgebra_de_Boole) y del [sistema binario](https://es.wikipedia.org/wiki/Sistema_binario) en el diseño de este tipo de circuitos y en su programación.

( <https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html> )

4° Lenguaje de bajo nivel (Ensamblador): Es un lenguaje de programación que se usa en los microprocesadores. Implementa una representación simbólica de los [códigos de máquina](https://es.wikipedia.org/wiki/Lenguaje_de_m%C3%A1quina) binarios y otras constantes necesarias para programar una [arquitectura de procesador](https://es.wikipedia.org/wiki/Arquitectura_de_CPU) y constituye la representación más directa del [código máquina](https://es.wikipedia.org/wiki/Lenguaje_de_m%C3%A1quina) específico para cada arquitectura legible por un programador. Un ensamblador es un programa que ejecuta la computadora y convierte instrucciones en un patrón de bits. El procesador puede usarlo para realizar operaciones básicas. Algunas personas llaman a estas instrucciones lenguaje ensamblador y otras usan el término lenguaje ensamblador*.*

( <https://es.itpedia.nl/2019/11/11/wat-is-een-assembler-taal/> )

5° Lenguaje de Alto Nivel: Lenguaje de programación que utiliza palabras clave e instrucciones similares a las expresiones del lenguaje humano o las matemáticas y que, por tanto, resulta más fácil de comprender y usar para las personas. Un lenguaje de alto nivel debe su nombre al alto nivel de abstracción que ofrece a los desarrolladores respecto a funciones de procesadores de bajo nivel, tales como acceso a memoria y almacenamiento de registro, por lo que tales operaciones no requieren la atención del desarrollador.

De igual forma, dichos lenguajes son prácticos y fáciles de portar lo cual simplifica el uso del programa y programador a la hora de solucionar problemas, como características de herencia y gestión autómata de memoria al igual que su interfaz grafica

( <https://muytecnologicos.com/diccionario-tecnologico/lenguaje-de-alto-nivel> )

6°Algormitmos cualitativos: Los algoritmos cualitativos son aquellos que se realizan por medio de las palabras, lo que quiere decir que las órdenes vienen dadas en forma verbal. Por ejemplo, una receta de cocina. Los algoritmos cualitativos se emplean con frecuencia en la vida cotidiana para resolver problemas. Por ejemplo: las instrucciones de uso que traen los equipos electrónicos, las instrucciones para el montaje de un equipo, las técnicas de laboratorio para evaluar ácidos, etc.

( <https://www.lifeder.com/algoritmos-cualitativos/> )

7° Algoritmos cuantitativos: Los algoritmos cuantitativos son aquellos algoritmos que utilizan operaciones algebraicas y cálculos numéricos específicos para definir un proceso, obteniendo valores concretos. Por ejemplo, el resultado de una resta o una multiplicación.

En las ciencias de la computación, en las matemáticas y otras disciplinas afines, un algoritmo es un conjunto finito y ordenado de instrucciones que permite efectuar una actividad por medio de pasos sucesivos que no generan dudas a quien deba ejecutar estas acciones, llevando a la solución de un determinado problema

( <https://www.studocu.com/ec/document/universidad-estatal-del-sur-de-manabi/informatica/tarea-2-concepto-y-ejercicios-de-1algoritmos-cualitativos-2algoritmos-cuantitativos/31516977> )

8° Lenguaje algorítmicos gráficos: Es una serie de símbolos y reglas que se utilizan para describir de manera grafica un problema, el cual se usan recursos como diagrama de flujo, diagramas estructurados, pseudocodigo.

( <https://jorgesuaza76.wixsite.com/facultaddeingenieria/lenguajes-algoritmicos> )

9° Lenguajes no gráficos: Se representa de forma descriptiva las operativas que se debe realizar un algoritmo en base de las operaciones digitales y mecánicas de un algoritmo.

10° Pseudocodigo: es una forma de expresar los distintos pasos que va a realizar un programa, de la forma más parecida a un lenguaje de programación. Su principal función es la de representar por pasos la solución a un problema o algoritmo, de la forma más detallada posible, utilizando un lenguaje cercano al de programación. El pseudocódigo **no puede ejecutarse en un ordenador** ya que entonces dejaría de ser pseudocódigo, como su propio nombre indica, se trata de un código falso (pseudo = falso), es un código escrito para que lo entienda el ser humano y no la máquina.

Su principal característica es la de representar un **método que facilita la programación** y solución del algoritmo del programa. También se caracteriza por ser una forma de representación, fácil de utilizar y de manipular, que simplifica el paso del programa, al lenguaje de programación.

Otra característica que tiene el pseudocódigo es su **independencia al código en el que se va a escribir el programa,** proporcionando un método que facilita la posterior programación y la resolución del algoritmo del programa.

( <https://openwebinars.net/blog/que-es-pseudocodigo/> )

11° Definición del problema: En ciencia computacional teorética, un inconveniente extraño o inconveniente subliminal es una relación entre un conjunto de puntos y comas y un conjunto de soluciones. Un inconveniente abstracto deja establecer formalmente la relación deseada entre la dama de un algoritmo y su hombre. El análisis del inconveniente define la naturaleza, el alcance, las causas y las consecuencias concretos del tema de la campaña para determinar el mejor modo de abordar esa cuestión y qué que puede mudarse.

12° Análisis del problema: El propósito del análisis de un problema es ayudar al programador (Analista) para llegar a una cierta comprensión de la naturaleza del problema. Una buena definición del problema, junto con una descripción detallada de las especificaciones de entrada/salida, son los requisitos más importantes para llegar a una solución eficaz.

Para realizar el análisis del problema realizamos varios pasos:

* **Definición del problema**: Tenemos que entender y comprender la naturaleza del problema, tenemos que dominar los conceptos con los que estamos trabajando y conocer que operaciones hay que realizar para solucionar el problema y obtener la información de salida.
* **Especificación de los datos de entrada**: Hay que determinar qué datos de entrada necesitamos para resolver el problema.
* **Especificación de los datos de salida**: También hay que determinar los datos de salida que van a ofrecer la solución del problema.

( <https://plataforma.josedomingo.org/pledin/cursos/curso_cpp1/curso/u01/#:~:text=Definici%C3%B3n%20o%20an%C3%A1lisis%20del%20problema,de%20un%20problema%20dado%3A%20algoritmo>. )

13° Diseño del Algoritmo: Un **algoritmo** es un conjunto de acciones que especifican la secuencia de operaciones realizar, en orden, para resolver un problema. Los algoritmos son independientes tanto del lenguaje de programación como del ordenador que los ejecuta.

Las características de los algoritmos son:

* Un algoritmo debe ser preciso e indicar el orden de realización de cada paso.
* Un algoritmo debe estar definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.
* Un algoritmo debe ser finito. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.

14° Codificación: La codificación es un proceso mediante el cual se asignan claves numéricas a las respuestas de preguntas abiertas de un cuestionario. Este proceso se encuentra integrado a una etapa de procesamiento integral de la información, que incluye la captura, validación y explotación de la información.

( <https://www.cepal.org/sites/default/files/presentations/sistema-codificacion-automatizada.pdf> )

15° Compilación y Ejecución: Es una técnica para mejorar el rendimiento de sistemas de programación que compilan a bytecode, consiste en traducir el bytecode a código maquina nativo en tiempo de ejecución. La compilación en tiempo de ejecución se construye a partir de dos ideas anteriores relacionadas con los entornos de ejecución: la compilación del bytecode y la compilación dinámica.

16° Prueba y Depuración: La depuración y las [pruebas](https://aws.amazon.com/blogs/devops/tag/testing/) son procesos complementarios que garantizan que los programas de software funcionen como deben. Después de escribir una sección completa o parte de un código, los programadores realizan pruebas para identificar fallos y errores. Una vez encontrados los errores, los codificadores pueden comenzar el proceso de depuración y trabajar para librar al software de cualquier error.

La depuración es el proceso de encontrar y solucionar errores en el código fuente de cualquier software. Cuando un software no funciona tal y como se espera, los programadores de computadoras estudian el código para determinar por qué ocurren algunos errores. Con este objetivo, los programadores utilizan herramientas de depuración para ejecutar el software en un entorno controlado, comprobar el código paso a paso y analizar y solucionar el error.

( <https://aws.amazon.com/es/what-is/debugging/> )

17° Documentación:  La documentación de software se define como la **información enfocada en la descripción del sistema o producto**para quienes se encargan de desarrollarlo, implementarlo y utilizarlo. Por lo que también puedes determinar como una documentación de procesos. Además, esta documentación incorpora los aspectos de los manuales, como las funciones de ayuda, sus versiones en línea y demás.

Es importante destacar también que la documentación de software**o documentación de procesos puede indicar los datos relacionados con las notas de sesión** y comentarios de código del sistema.

Asimismo, este elemento **permite la comprensión del producto de**software**, su interfaz, sus capacidades, propiedades principales** y demás. Además, la documentación de software será clave para facilitar encontrar de manera rápida una determina sección.

( <https://keepcoding.io/blog/que-es-la-documentacion-de-software/#:~:text=La%20documentaci%C3%B3n%20de%20software%20se,como%20una%20documentacion%20de%20procesos>. )

18° Tipos de Datos: Un **tipo de datos** es la propiedad de un valor que determina su dominio (qué valores puede tomar), qué operaciones se le pueden aplicar y cómo es representado internamente por el computador. Todos los valores que aparecen en un programa tienen un tipo los cuales son:

* Tipo entero: Del tipo ( int ) permite representar números enteros, los valores enteros lineales se escriben con un signo opcional seguido por una secuencia de dígitos:
* Tipo reales: El tipo **float** permite representar números reales. El nombre float viene del término [punto flotante](http://es.wikipedia.org/wiki/Punto_flotante), que es la manera en que el computador representa internamente los números reales. Hay que tener mucho cuidado, porque los números reales no se pueden representar de manera exacta en un computador. Por ejemplo, el número decimal 0.7 es representado internamente por el computador mediante la aproximación 0.69999999999999996. Todas las operaciones entre valores float son aproximaciones. Esto puede conducir a resultados algo sorpresivos:

( <http://progra.usm.cl/apunte/materia/tipos.html> )

19° Expresiones: Las expresiones son secuencias de operadores y operandos que se utilizan para uno o más de estos propósitos:

* Calcular un valor a partir de los operandos.
* Designar objetos o funciones.
* Generar “efectos secundarios”. (Los efectos secundarios son cualquier acción distinta de la evaluación de la expresión; por ejemplo, modificando el valor de un objeto).

En C++, los operadores se pueden sobrecargar y el usuario puede definir sus significados. Sin embargo, la prioridad y el número de operandos que aceptan no pueden modificarse. En esta sección se describe la sintaxis y la semántica de los operadores tal como se proporcionan con el lenguaje, no sobrecargados. Además de los [tipos de expresiones](https://learn.microsoft.com/es-es/cpp/cpp/types-of-expressions?view=msvc-170) y la [semántica de las expresiones](https://learn.microsoft.com/es-es/cpp/cpp/semantics-of-expressions?view=msvc-170).

( <https://learn.microsoft.com/es-es/cpp/cpp/expressions-cpp?view=msvc-170> )

20° Operadores y operandos: Un "operando" es una entidad en la que actúa un operador. Una "expresión" es una secuencia de operadores y operandos que realiza cualquier combinación de estas acciones:

* Calcula un valor
* Designa un objeto o una función
* Genera efectos secundarios

Los operandos de C incluyen constantes, identificadores, cadenas, llamada de función, expresiones subíndice, expresiones de selección de miembro, y expresiones complejas que se forman combinando operandos con operadores o incluyendo operandos entre paréntesis. La sintaxis de estos operandos se proporciona en [Expresiones primarias](https://learn.microsoft.com/es-es/cpp/c-language/c-primary-expressions?view=msvc-170).

Las expresiones realizan acciones específicas, según un operador, con uno o dos operandos. Un operando puede ser una constante, una variable o el resultado de una función. Los operadores son aritméticos, lógicos y relacionales. Al igual que sucede en C, la funcionalidad de un operador puede variar según el tipo de datos de los operandos especificados en la expresión.

**Operadores aritméticos ( +, -, \*, /, \*\*, % )**

Los operadores aritméticos realizan operaciones matemáticas, como sumas o restas con operandos. Hay dos tipos de operadores matemáticos: unarios y binarios. Los operadores unarios realizan una acción con un solo operando. Los operadores binarios realizan acciones con dos operandos. En una expresión compleja (dos o más operandos), el orden de evaluación depende de las reglas de precedencia.

**Operadores aritméticos unarios**

Los operadores unarios son operadores aritméticos que realizan una acción sobre un solo operando. El lenguaje de script reconoce el operador unario negativo (-).

El operador unario negativo invierte el signo de una expresión, de positivo a negativo o viceversa. El efecto neto es el de multiplicar el número por -1.

( <https://www.ibm.com/docs/es/iad/7.2.1?topic=language-operators-expressions> )

( <https://learn.microsoft.com/es-es/cpp/c-language/operands-and-expressions?view=msvc-170> )

21° Identificadores: Son elementos textuales llamados símbolos que nombran entidades del lenguaje. Algunas entidades que un identificador puede denotar son las variables, las constantes, los tipos de datos, las etiquetas, las subrutinas y los paquetes; Un constante es un dato cuyo valor no puede cambiar durante la ejecución del programa, de igual forma es nombrado asociado a un elemento de datos que esta situado en posiciones contiguas de la memoria principal y su valor puede cambiar durante la ejecución del programa

( <https://sites.google.com/a/misena.edu.co/wiki-adsi/home/identificadores-variables-y-constantes> )

22° Dispositivos de entrada: Los distintos dispositivos de entrada**traducen los datos en impulsos eléctricos**, que luego son transmitidos a la computadora para su procesamiento y almacenamiento en la memoria central o interna. Estos dispositivos son fundamentales porque permiten que el usuario pueda hacer uso de la computadora e interactuar con ella.

Los dispositivos de entrada se distinguen de los dispositivos de salida, que son aquellos aparatos que se conectan a una computadora y extraen información.

Además, **algunos dispositivos de entrada lo son también de salida** (dispositivos de E/S o mixtos) porque introducen y extraen información del sistema. Por ejemplo: un monitor táctil.

( <https://concepto.de/dispositivos-de-entrada/> )

23° Dispositivos de Salida: Para que el usuario pueda hacer uso de **la computadora necesita tanto de los dispositivos de entrada como de los de salida.** Los [dispositivos de salida](https://concepto.de/dispositivos-de-salida/) son aquellos que traducen los datos provenientes de la computadora en información que pueda ser comprendida por el usuario.

En la mayoría de los casos, esta información se presenta en forma de imagen, sonido, texto, entre otras. Los dispositivos de salida más importantes son: la pantalla o monitor, los parlantes y la impresora.

Además, existen algunos dispositivos que funcionan tanto como dispositivos de entrada (porque introducen información al sistema) como de salida (porque extraen información de la computadora). Estos dispositivos son llamados de entrada/salida o mixtos y entre los más destacados están:

* **Impresoras multifunción**. Son dispositivos que cuando imprimen funcionan como dispositivo de salida y cuando escanean funcionan como dispositivo de entrada.
* **Monitores táctiles**. Son dispositivos que funcionan a la par como de entrada y salida porque muestran la información al usuario y también permiten que la persona ingrese directivas al pulsar la pantalla.
* **Módems**. Son dispositivos que convierten las señales digitales en analógicas (y viceversa) y permiten la conexión de un dispositivo a internet.

24° Unidad Central De Procesamiento (CPU): CPU son las siglas de Central Processing Unit, lo que traducido significa Unidad Central de Procesamiento. Se trata de uno de los componentes vitales que te vas a encontrar en tu ordenador, tu smartphone o tu tableta o portátil, e incluso en relojes y prácticamente cualquier dispositivo electrónico. Sin él, simplemente no podrían funcionar.

A la CPU se la suele llamar coloquialmente como microprocesador o simplemente procesador, y puedes considerarla como **el cerebro de cualquier dispositivo**. Se encarga de procesar todas las instrucciones del dispositivo, leyendo las órdenes y requisitos del sistema operativo, así como las instrucciones de cada uno de los componentes y las aplicaciones.

( <https://www.xataka.com/basics/cpu-que-como-sirve> )

25° Unidad de control: La unidad de control es el componente del procesador que dirige y coordina la mayoría de las operaciones en la computadora. La unidad de control tiene un rol mucho muy parecido al que tendría un oficial de tránsito vehicular ya que está se encarga de interpretar cada una de las instrucciones generadas por un programa y después inicia las acciones apropiadas para llevar a cabo las instrucciones. Los tipos de componentes internos que la unidad de control dirige incluyen la unidad lógica y aritmética, los registros, y los buses*.*

( <http://cca.org.mx/cca/cursos/cucfc/modulo2/tema2-02.html#:~:text=La%20unidad%20de%20control%20es,las%20operaciones%20en%20la%20computadora>. )

26° Unidad aritmético-lógica: La unidad aritmético lógica, también conocida como ALU (siglas en inglés de arithmetic logic unit), es un circuito digital que calcula operaciones aritméticas (como suma, resta, multiplicación, etc.) y operaciones lógicas, entre dos números. así como establece comparaciones lógicas a través de los condicionales lógicos “si”, “no”, y, “o”. Desde los circuitos más simples, como relojes y calculadoras, hasta complejos circuitos, como los microchips actuales, todos incluyen al menos una Unidad aritmético-lógica, que varía su poder y complejidad según su finalidad.

( <https://www.ecured.cu/Unidad_Aritm%C3%A9tico_L%C3%B3gica> )

27° Memoria central: Es la memoria de la [computadora](https://es.wikipedia.org/wiki/Computadora) donde se almacenan temporalmente tanto los datos como los programas que la [unidad central de procesamiento](https://es.wikipedia.org/wiki/Unidad_central_de_procesamiento) ([CPU](https://es.wikipedia.org/wiki/Central_Processing_Unit)) está procesando o va a procesar en un determinado momento. Por su función, la Memoria debe ser inseparable del [microprocesador](https://es.wikipedia.org/wiki/Microprocesador) o CPU, con quien se comunica a través del [bus de datos](https://es.wikipedia.org/wiki/Bus_de_datos) y el [bus de direcciones](https://es.wikipedia.org/wiki/Bus_de_direcciones). El ancho del [bus](https://es.wikipedia.org/wiki/Bus_(inform%C3%A1tica)) determina la capacidad que posea el microprocesador para el direccionamiento de direcciones en memoria.

28° Memoria Auxiliar: **Memorias Secundarias**, que también son conocidas como **Memoria Auxiliar**, que a diferencia de la **Memoria**[RAM](https://sistemas.com/ram.php), por ejemplo, no tienen un almacenamiento temporal de los **Datos** procesados, sino que permiten guardarlos en forma permanente, inclusive cuando desconectamos el ordenador de su alimentación de**Energía Eléctrica**, por lo que se puede afirmar entonces que se trata de un **Almacenamiento Permanente** de esta información.

Está conformada entonces por una gran variedad de **Medios de Almacenamiento** que permiten el guardado de una gran cantidad de datos, diferenciable con las otras memorias no solo en lo que respecta a esta cualidad, sino también en su calidad, ya que la capacidad de almacenamiento es ampliamente superior.

Su funcionamiento no es autónomo, ya que justamente trabajan en conjunto con la **Memoria Primaria**, por lo que, por ejemplo, para la lectura de un archivo en un **Medio de Almacenamiento**, es posible que se requiera de la utilización de la **Memoria RAM** para agilizar esta lectura e incrementar el rendimiento de los procesos.

( <https://sistemas.com/memoria-auxiliar.php> )

29° Memoria RAM: es una pequeña memoria de rápidas prestaciones que usa el ordenador para trabajar internamente y almacenar datos a los que necesita acceso rápido. Lo de acceso aleatorio quiere decir que puedes acceder a cualquier parte de la memoria instantáneamente sin tener que leer primero las anteriores. El otro tipo de memoria es la llamada secuencial, que evidentemente es mucho más lenta, pues antes de leer esa tercera giga debe acceder al primero y al segundo. Imaginaos una cinta de audio o un CD, que gira sobre sí mismo y avanza y retrocede para leer una u otra parte. En la [memoria RAM](https://sistemas.com/memoria-ram.php) el acceso es inmediato, no tiene que avanzar o retroceder, sino que va directo al sitio en el que se le indica que está la información.

( <https://sistemas.com/ram.php> )

30° Memoria ROM: En [informática](https://concepto.de/informatica/), cuando hablamos de memoria ROM (acrónimo de Read–Only Memory, es decir, Memoria de Sólo Lectura), nos referimos a**un tipo de almacenamiento empleado en**[computadores](https://concepto.de/computador/)**y otros dispositivos electrónicos**, que se caracteriza por ser únicamente de acceso para lectura y nunca para escritura, es decir, que se la puede recuperar pero no modificar o intervenir.

La memoria ROM es de acceso secuencial y su presencia es independiente de la presencia de una [fuente de energía](https://concepto.de/fuentes-de-energia/). Como se ha dicho,**su contenido no puede modificarse**, o al menos no de manera simple y cotidiana, y suele contener [información](https://concepto.de/informacion/) introducida en el sistema por el fabricante, de tipo básico, operativo o primario.

( <https://concepto.de/memoria-rom/> )